Team:
Tarun Singh
Vennela Gandluri
Rohith Sali

GitHub names:
Tarun Singh (TarunSinghD)
Vennela Gandluri (vega5334)
Rohith Sali (Rohithsali)

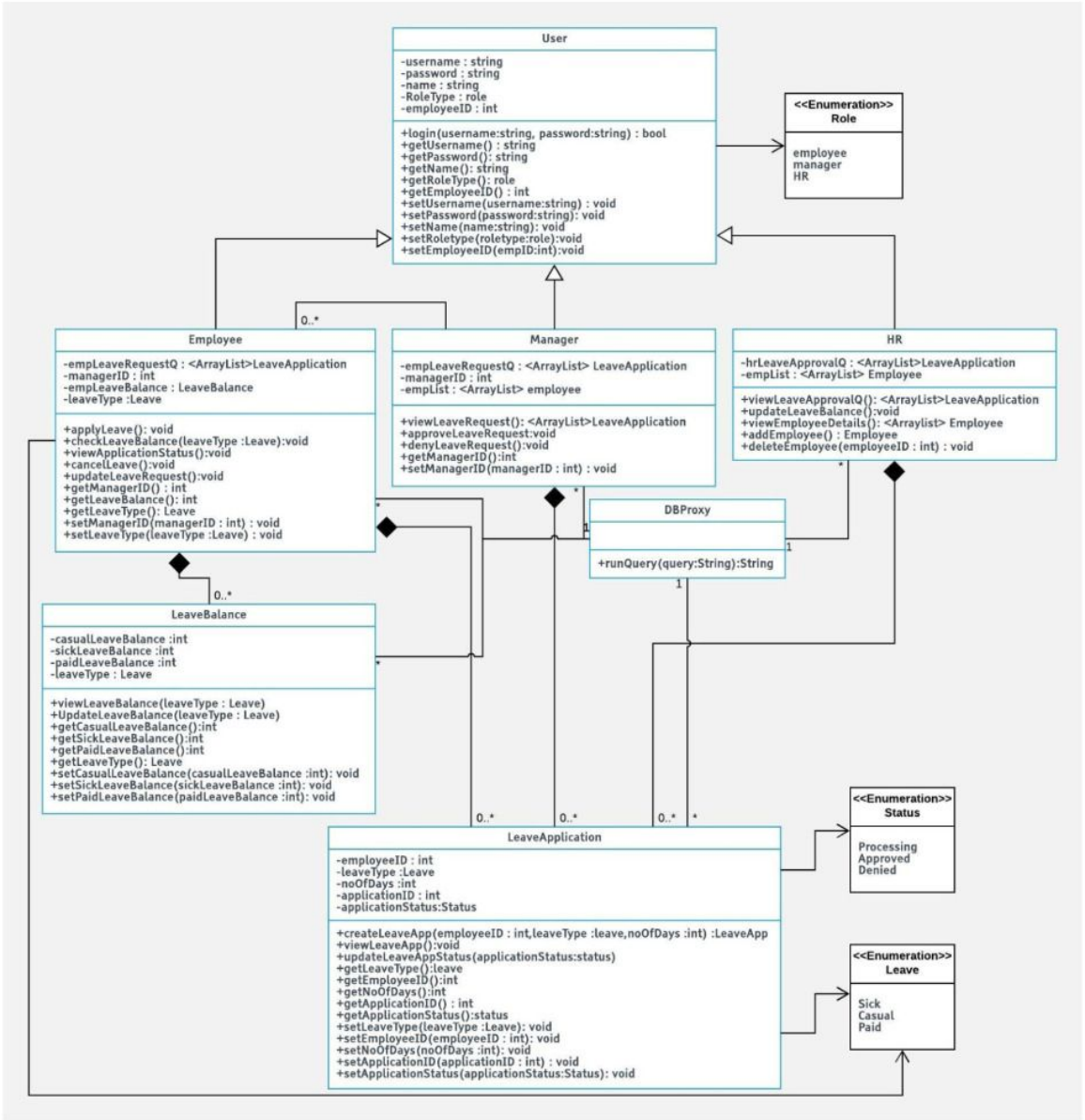
Project Title: Leave Management System

Description:
A company has a large number of employees which includes full/part time employees and contractors. To smoothly manage the leave tracking operations of the company employees and to make the process error free and faster, the management has come up with a proposal to create an online, web based application called LMS (Leave Management System). Being an online application, LMS can be accessed through internet round the clock and is also an environment friendly way to avoid paper based processing of employee leave data and leave approval tracking details. The new system will greatly reduce the amount of time spent in leave processing because company's present system of manual tracking is very slow.

Previous class diagram (From part 2):

Link:https://github.com/TarunSinghD/Object-Oriented-Design-Final-Project/blob/master/Class_Diagram_Part2.jpeg

Class Diagram



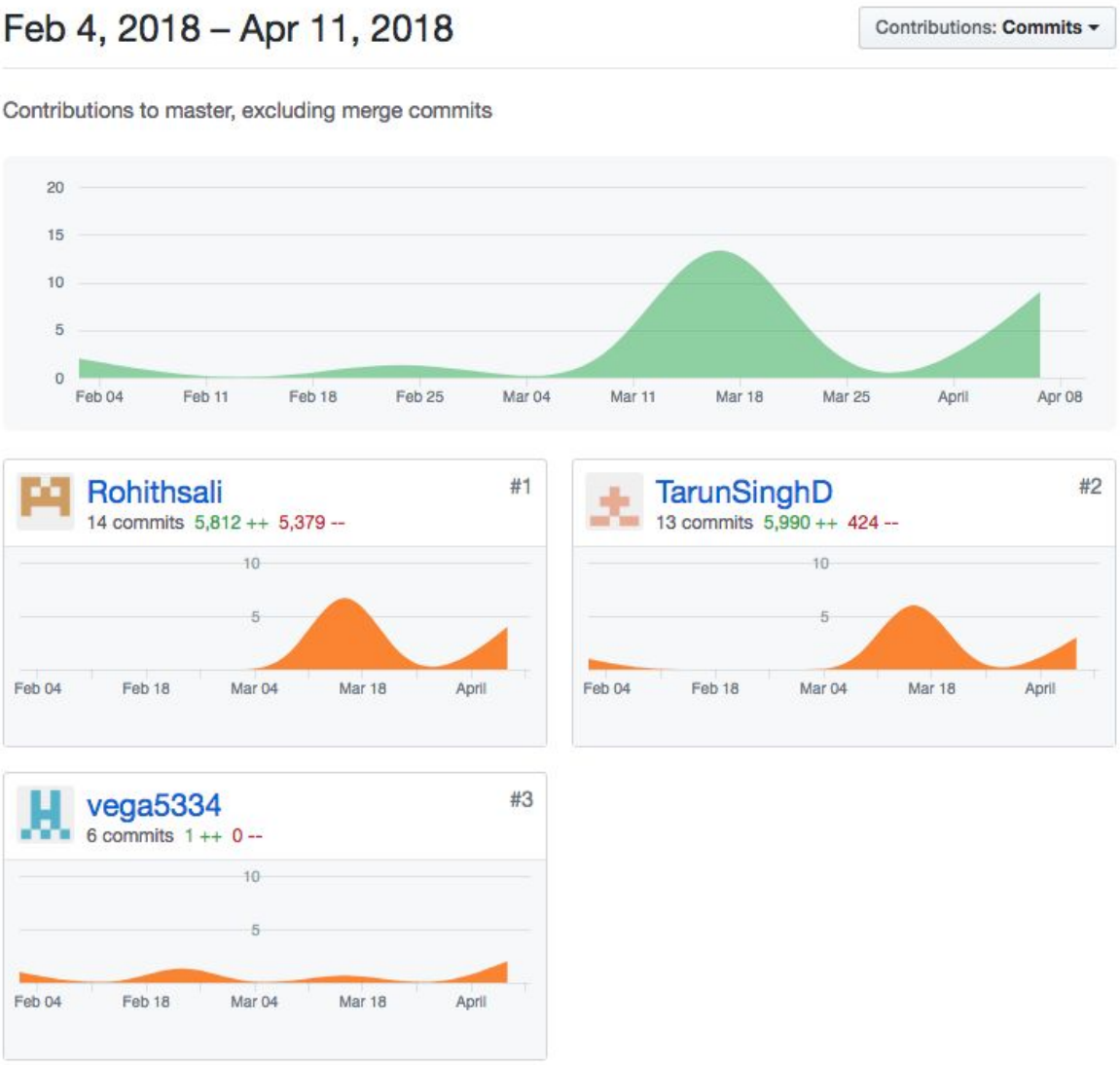
Summary of work done:
Most of the effort in the last two weeks has been aimed towards creating database using mySQL. Webpage (using HTML) creation linking it to the java project, and connecting it to the tomcat server was a main focus.

Breakdown of work:
-Creation of database using mySQL: Rohit
-Creation of web pages and linking them to Java using Java servlets: Tarun, Rohit, Vennela
-Design pattern implementation in class diagram.

1. Observer- Tarun
2. Factory-Vennela
3. Singleton- Rohit

Estimated remaining effort:
Tasks pending are:
-We have decided to implement the web pages and database on spring boot instead of html and java servlets.
-Implementation of all 3 design patterns

Github Graph:



Design patterns:

1.Factory design pattern:

We have chosen this design pattern to use polymorphism on User object - which can be instantiated to either Employee, Manager or HR by a factory class based on the role type. The following changes have been made to the class diagram with respect to this design pattern:

- Factory method UserFactory has been added.

2. Observer design pattern

The Leave Application object has a “status” field associated with it.
Employee should be notified when the status is changed to either approved or rejected.
Manager should be notified when a “new” application is created.

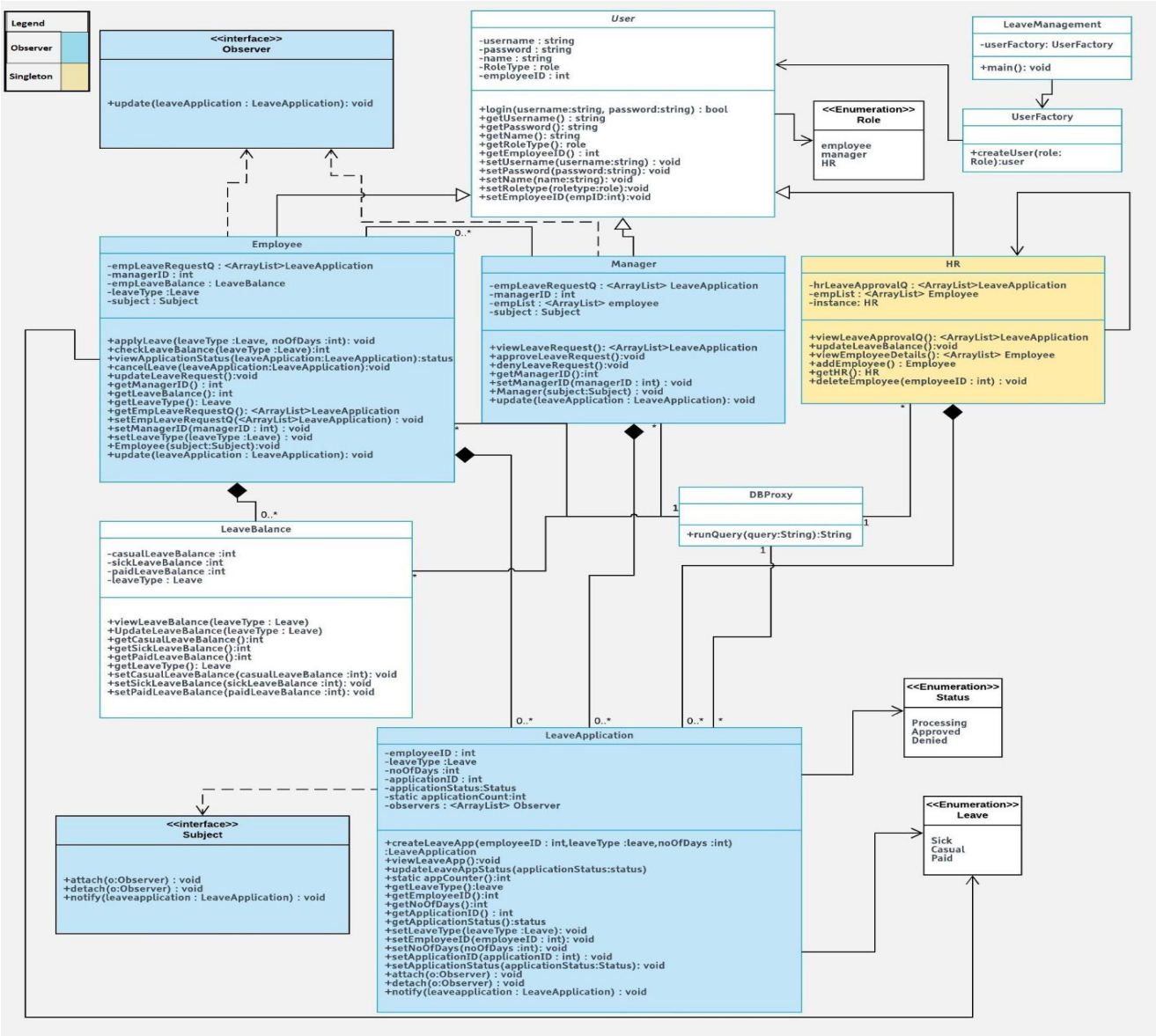
The observer design pattern fits well here with the publish-subscribe model. The manager and employee objects subscribe to any changes in the status field.

3. Singleton design pattern

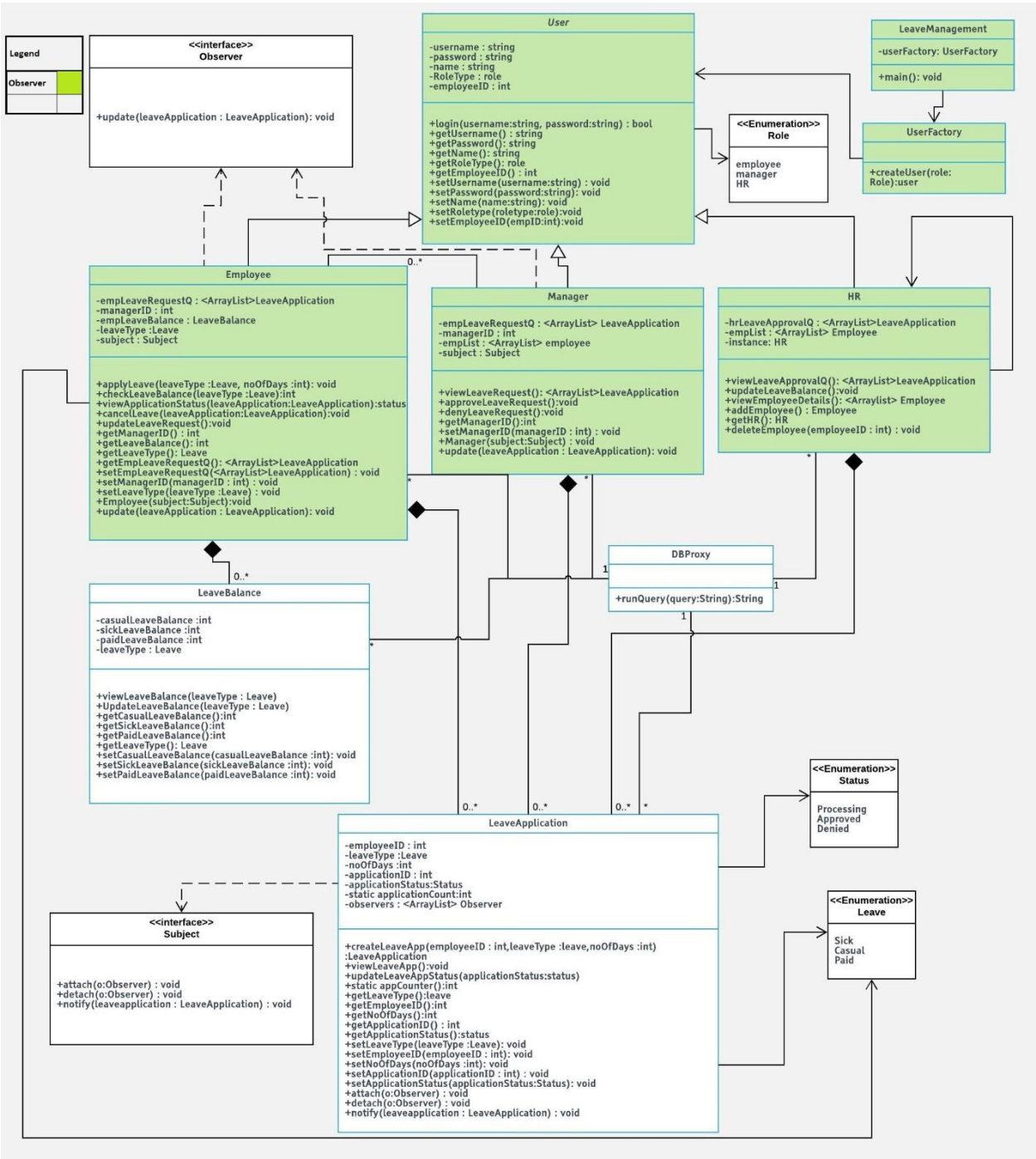
This pattern was implemented so that there is only one HR for the entire system. So the HR class can be instantiated only once.

Class diagram with design patterns

Class diagram with singleton and observer patterns highlighted:



Class diagram with Factory pattern highlighted:



Final Iteration:

- Implement the three design patterns

- Complete implementation of database and UI design using spring boot.